Logic in Cognitive Science: Bridging the Gap between Symbolic and Connectionist Paradigms

Logic in Cognitive Science: Bridging the Gap between Symbolic and Connectionist Paradigms

Alistair Isaac[1] and Jakub Szymanik[2]

[1]Department of Philosophy, Stanford University
[2]Department of Philosophy, Stockholm University

July 30, 2010

1 Introduction

What can logic contribute to cognitive science? In the early days of cognitive science, logic was taken to play both a descriptive and a normative role in theories of intelligent behavior. Descriptively, human beings were taken to be fundamentally logical, or rational. Normatively, logic was taken to define rational behavior and thus to provide a starting point for the artificial reproduction of intelligence. Both positions were soon challenged. As it turns out, however, logic continues to be at the forefront of conceptual tools in cognitive science. Rather than defeating the relevance of logic, the challenges posed by cognitive science have inspired logicians to enrich the repertoire of logical tools for analyzing reason, computation, and communication. After a brief survey of the wide array of roles played by logic in cognitive science, we will examine the role of non-monotonic logics and complexity theory in more detail.

Classical logic provides the groundwork for the abstract theory of computation, which in turn was used by Alan Turing (1950) to define the challenge of human level artificial intelligence. Turing proposed that the true test of machine intelligence is indistinguishability from human intelligence and suggested a concrete method for determining if this criterion is satisfied. A human judge sits in front of two terminals, one allows him to communicate with a computer and the other one with a human being. The judge can type whatever text he chooses into the terminals. His task is to use the answers he receives to decide which terminal is connected to a human and which to a machine. If the judge cannot tell the difference, the computer has passed the Turing test. The Loebner Prize for artificial intelligence is an annual competition which awards cash prizes to the computers which come closest to passing the Turing test.

Turing's test is a natural extension of the view that the human mind is a computational device. It assumes the Church-Turing thesis, which states that all computation (in the intuitive sense of a mechanical procedure for solving problems) is formally equivalent to Turing computation (computation by a Turing machine). This is a conceptual claim which cannot be formally proved. However, all attempts so far to explicate computability (many of them independently motivated) have turned out to define exactly the same class of problems. For instance, definitions via abstract machines (random access machines, quantum computers, cellular automata, genetic algorithms), formal systems (the lambda calculus, Post rewriting systems), and particular classes of function (recursive functions) are all formally equivalent to the definition of Turing machine computability. These results provide compelling support for the claim that all computation is equivalent to Turing computation (see e.g. Cooper, 2003, or his paper in this volume for more details).

If the Church-Turing thesis is correct, then all computations performed by the human brain could, in theory, be reproduced on any device equivalent to a Turing machine, e.g. a modern computer (modulo sufficient time, memory, and processing power). The artificial reproduction of intelligence should be easy, then: all we need to do is analyze the function being computed and program it into our computer. However, as it turns out, things are not so simple. A straightforward analysis of the steps needed to perform a complex task works well for reproducing intelligent behavior within narrowly defined environments, such as the game of chess. Reasoning about complex general domains has turned out to be very difficult to reproduce, however, as will be discussed in more detail below.

One strategy for simplifying the problem is to focus, not on intelligence tout court, but rather some more narrowly defined skill which contributes to human reasoning. Language, for example, plays a crucial role in human cognition, yet the transparency of its combinatorial structure makes it a natural target for formal models. Since we reason with language, analysis of semantic structure can often lead to insights into reasoning strategies. For example, van Lambalgen and Hamm (2004) begin with an algorithmic conception of meaning (Suppes, 1982; Moschovakis, 2006) and extend this to an analysis of temporal reasoning. Likewise, studies of the semantics (Clark, 1976; Moxey and Sanford, 1993; Peters and Westerståhl, 2006) and pragmatics (Geurts, 2010) of quantifiers have contributed to our understanding of reasoning about quantities, especially as expressed in syllogisms (Geurts, 2003; Johnson-Laird, 2008). When supplemented with computational analysis, these logical models can deliver concrete psychological (Szymanik and Zajenkowski, 2010; Gierasimczuk and Szymanik, 2009) and even neuroanatomical (Szymanik, 2007; Clark and Grossman, 2007) predictions.

The power of logical methods in cognitive science is not limited to

modeling language, however. Logic has motivated models of neural behavior, both historically (the classic work of McCulloch and Pitts, 1943) and today (e.g. Sandler and Tsitolovsky, 2008). At a more abstract level of cognitive organization, logic can analyze constraints on concept learning (e.g. Feldman, 2000) and the acquisition of a Theory of Mind in children (Verbrugge and Mol, 2008; Flobbe et al., 2008). After concepts have been acquired, modal and epistemic logics can provide models of knowledge representation (van Harmelen et al., 2008) and of higher-order epistemic reasoning and social cognition (Verbrugge, 2009). So, logic has emerged as a powerful modeling tool at every level of cognitive abstraction. It constitutes one of the common languages with which researchers in the cognitive sciences can communicate across disciplinary boundaries.[1]

In fact, the very success of logic in analyzing intelligent behavior has motivated questions at the interface between experimental and theoretical approaches to understanding cognition. For example, a much repeated experiment, the Wason (1968) selection task, seems to indicate that humans are very poor at applying even simple rules of reasoning such as modus tollens. Furthermore, Cheng et al. (1986) suggests they may continue to be poor even when they have taken an introductory logic class! Does this imply that humans are not fundamentally rational? Or that our neural wiring is somehow qualitatively different from the logical structure which can be found in any computational device simulating the brain? What is the relation here between the brain's computational processes and the logic which governs them?

One theoretical response to challenges such as these is to abandon the logical paradigm in favor of neural networks, computational models inspired by the physiology of the brain. Although the 1980's and '90's saw fierce debate between supporters of the neural network, or connectionist, paradigm and those of the logical, or symbolic, paradigm, recent results indicate that at a deep formal level the two types of model are computationally equivalent. In particular, a large class of neural network models can be interpreted as performing calculations in accordance with the rules of non-monotonic logic. Computational equivalence does not imply practical equivalence, however. Here again the resources of classical logic, in the form of complexity theory, can provide insight. We argue that choice of connectionist or symbolic paradigm should not influence the analysis of cognitive tractability. Therefore, at the computational level, task analysis in both paradigms should converge.

In the following section, we briefly survey the development of non-monotonic logics within artificial intelligence, including both successes and

[1] For more references on the interface between logic and cognition, see also the 2007 special issue of *Topoi* on "Logic and Cognition", edited by Johan van Benthem and Helen and Wilfrid Hodges, and the 2008 special issue of *Studia Logica* on "Psychologism in Logic?", edited by Hannes Leitgeb.

stumbling blocks. Next, we will examine the application of recent results on the relationship between non-monotonic logics and neural networks to the philosophical debate between connectionist and symbolic paradigms in cognitive science. Section 4 introduces the basic principles and results of complexity theory, while Section 5 demonstrates how they can be used to analyze the computational plausibility of cognitive models. Here we introduce the concept of tractability: a tractable problem is one which can be solved efficiently. Computational complexity theory implies that precisely the same problems are tractable for both connectionist and symbolic models. Strong assumptions must be made about contingent features of the model (number of parallel channels for computation) or the environment (e.g. statistical regularities) in order for any differences in computational power to emerge. Therefore, there is no difference between symbolic and connectionist models at the level of what they can accomplish *in principle*; furthermore, even when analyzing cognition *in practice* (e.g. in an experimental setting), tractability concerns can guide the interpretation of results.

2 Non-monotonic Logic and Human Reasoning

A function f is said to be *monotonic* if $n \leq m$ implies $f(n) \leq f(m)$; essentially, as the input grows, the output grows as well. Reasoning in classical logic is monotonic because adding new premises always allows you to generate more conclusions. Let T and T' represent consistent sets of sentences and let $F(T)$ denote the deductive closure of T (i.e. the set of all sentences which follow from T by some specified (classical) inferential rules). Then, for all classical logics, $T \subseteq T'$ implies $F(T) \subseteq F(T')$.

Typically, a non-monotonic logic supplements an underlying classical logic with a new, non-monotonic connective and a set of inference rules which govern it. The rules describe a logic of *defeasible* inference, inferences which are usually safe, but which may be defeated by additional information. For example, from the fact that *this is a bird*, I can usually conclude that *this can fly*. This inference can be defeated, however, if I learn that *this is a penguin*. Symbolically, we want our system to ensure that $Bird(x) \Rightarrow Fly(x)$, but $Bird(x) \wedge Penguin(x) \not\Rightarrow Fly(x)$. Incidentally, this example also demonstrates why such a system is non-monotonic, since $\{Bird(x)\} \subset \{Bird(x), Penguin(x)\}$ yet $F(\{Bird(x)\}) \not\subset F(\{Bird(x), Penguin(x)\})$. Non-monotonic rules of inference go by a variety of names, including circumscription (McCarthy, 1980), negation as failure (Clark, 1978), and default reasoning (Reiter, 1980).

Kraus et al. (1990) provide a unified approach to a hierarchy of non-monotonic logics of varying strengths. They distinguish each logic in this hierarchy in terms of the inference rules satisfied by their respective non-monotonic connectives. Consider, for example, the rule *Loop*:

$$\frac{\alpha_0 \Rightarrow \alpha_1, \alpha_1 \Rightarrow \alpha_2, \ldots, \alpha_{k-1} \Rightarrow \alpha_k, \alpha_k \Rightarrow \alpha_0}{\alpha_0 \Rightarrow \alpha_k} \qquad (\textit{Loop})$$

It should be obvious that a connective $\Rightarrow$ which satisfies *Loop* need not be as strong as the material conditional of classical logic. The material conditional satisfies transitivity ($A \rightarrow B$ and $B \rightarrow C$ imply $A \rightarrow C$), and *Loop* is an immediate consequence of transitivity (while the converse is not true). However, *Loop* is not satisfied in the weakest system considered by Kraus et al., which they call **C** for cumulative reasoning. The system **CL**, or cumulative reasoning with *Loop*, is an example of a non-monotonic reasoning system which is strictly weaker than classical logic, yet stronger than the weakest systems of non-monotonic reasoning.

It is unsurprising that non-monotonic logics should turn out to be useful in cognitive science. As early as 1980, John McCarthy argued that "humans use . . . 'non-monotonic' reasoning and . . . it is required for intelligent behavior" (McCarthy, 1980, p. 28). McCarthy's investigation of non-monotonic reasoning was motivated by the failure of A.I. to extend its success in limited domains (such as chess) to the "common sense" world in which humans reason so effectively. In a complex domain, humans are able to reason swiftly and effectively about both those features of the world which change *and those which do not* when some event occurs. The problem of how to keep track of those features of the world which do not change is called the "Frame Problem" (McCarthy and Hayes, 1969).

The Frame Problem comes in both a narrow and a broad version (see discussion in Dennett, 1984). The broad version concerns the potential relevance of any piece of information in memory for effective inference. Philosophers of cognitive science have worried about this broad problem since at least Fodor (1983). The narrow problem, however, concerns keeping track of stability in a changing world. This problem is effectively solved by non-monotonic logic; for a complete history and detailed treatment, see Shanahan (1997).

But how do human beings solve the Frame Problem? If we employ the same methods that worked in artificial intelligence, then our high-level reasoning should rest upon a substratum of simple logical procedures. However, when asked to perform very simple logical tasks, humans do surprisingly poorly. An elegant and compelling illustration of this can be found in the *Wason Selection Task* (Wason, 1968; Wason and Shapiro, 1971). The original Wason selection task is very simple. Subjects are shown four cards and told that all cards have numbers on one side and letters on the other. The faces visible to the subject read D, K, 3, and 7. The subject is then told "Every card which has a D on one side has a three on the other" and asked which cards they need to turn over to verify this rule. From a classical standpoint, the claim has the basic structure of a material conditional, $D \rightarrow 3$, and the correct answer is to turn over cards D and 7. However, the most popular answers (in order of decreasing popularity) are (1) D and 3;

(2) D; (3) D, 3, and 7; (4) D and 7. The classically correct answer ranks fourth, while an instance of affirming the consequent (i.e. judging that 3 is relevant for determining if the rule is correct) ranks first. Wason's robust and easily reproducible results seem to show that most people are poor at modus tollens and engage in fallacious reasoning on even very simple tasks. Are we really this bad at logic?

As it turns out, the story is more complex than this. The original selection task involved an abstract domain of numbers and letters. When the problem is rephrased in terms of certain types of domain with which subjects are familiar, reasoning suddenly improves. For example, Griggs and Cox (1982) demonstrate that if cards have ages on one side and types of drink on the other, subjects perform near perfectly (i.e. in accordance with the classical recommendation) when the task is to determine which cards to turn over to ensure that the rule "if a person is drinking beer, then that person is over 19 years old" is satisfied. This study builds upon earlier work by Johnson-Laird et al. (1972), demonstrating a similar phenomenon when the task involves postal regulations.

Johnson-Laird et al. (1972) and Griggs and Cox (1982) conclude that humans are better at logical reasoning in domains with which they are familiar. Since the original Wason task involves an abstract domain of letters and numbers, subjects are confused and fail to reason correctly. Cosmides (1989) and Cosmides and Tooby (1992) expand on these results and argue that they tell us something about cognitive architecture. In particular, Cosmides and Tooby conjecture that questions about postal regulations, drinking laws, etc. trigger a "cheater detection module". This module is hard wired to reason effectively, but in the domain-general case (when cheating may not be involved), we have no innate tendency to behave logically.

The most recent work on the logical analysis of the Wason selection task is a collaboration between psychologist Keith Stenning and logician Michiel van Lambalgen (2008). They point out that Wason's assertion that there is only one correct answer to the task is too quick, as it assumes the question is interpreted as stating a material conditional. Subjects who interpret the question as stating some other kind of dependency between D's and 3's than that captured by the material conditional are not necessarily making an error. The key here is in figuring out the relevant difference between versions of the task on which subjects perform in accordance with classical rules and versions (such as the original) on which they do not. Is it because the latter are abstract and the former concrete? Because the former rules are deontic and the latter are not? Stenning and van Lambalgen's novel suggestion here is that the crucial difference is in whether the subject sees the task as merely checking satisfaction of instances or as actually determining the truth of a rule. In the case of familiar deontic rules, their truth is not at issue, only whether or not they are being satisfied. The deontic nature of these rules means that turning cards over cannot falsify them (i.e. underage

drinking is still wrong, even if one discovers that it occurs), and this strictly limits interpretation of the task to checking the rule has been satisfied. In contrast, the original version of the task may be interpreted as involving either a descriptive or a prescriptive rule, greatly increasing the cognitive burden on the subject.

The ultimate analysis of the Wason selection task by Stenning and van Lambalgen is too sophisticated to give here. In broad outline, they first distinguish the interpretation step from the processing step in solving a selection task. One must first interpret the question being asked, only then can one compute a solution. Second, on the processing side, they argue that non-monotonic logic provides an appropriate model for analyzing subjects' behavior on this and related tasks. What level of mental processing corresponds to symbolic non-monotonic inference here? The answer of Lambalgen and Stenning may be surprising to the philosopher, as they prove a formal equivalence between symbolic and neural network models.

3 Symbolic vs. Connectionist Paradigms

In the late 1980's and early '90's, there were a sequence of heated debates between those advocating a symbolic and those advocating a connectionist (i.e. neural network based) approach to cognitive science. Perhaps the most famous of these is that initiated by Fodor and Pylyshyn (1988), which argued that (i) mental representations exhibit systematicity; (ii) representations in neural networks do not exhibit systematicity; therefore (iii) the appropriate formalism for modeling cognition is symbolic (not connectionist). Systematicity here is just the claim that changes in the meaning of a representation correspond systematically to changes in its internal structure (e.g. from my ability to represent "John loves Mary", it follows that I can also represent "Mary loves John"). Fodor and Pylyshyn (1988) claim that the only case in which representations in a neural network do exhibit systematicity is when the network is a mere implementation of a symbolic system (although they do not indicate how such implementational networks avoid their general critique, see Chalmers, 1990).

In the ensuing debate, much discussion focused on the special representational properties of neural networks; in particular, their use of "distributed", or "subsymbolic", representations. Smolensky (1987, 1988, 1991), van Gelder (1990, 1991), Clark (1993), and many others all emphasize the importance of acknowledging the distinctive properties of distributed representations in understanding the difference between neural networks and symbolic systems. Yet it is difficult to put one's finger on what the essential feature of a distributed representation is which makes it qualitatively different from a symbolic representation. Since the 1990's, hybrid models have risen to prominence (e.g. the ACT-R architecture of Anderson and Lebiere,

1998, or the analysis of concepts in Gärdenfors, 2000). These hybrid models combine neural networks (for learning) and symbolic manipulation (for high-level problem solving). Although pragmatically satisfying, the hybrid approach avoids rather than resolves any questions about the essential difference between symbolic and distributed representations.

The close relationship between classical logic and symbolic computation by Turing machines is well known (see e.g. Immerman, 1999, or the paper by Ramanujam in this issue). But what about subsymbolic computation by neural networks over distributed representations? With logic, we can show an essential equivalence between a logical system and neural networks if we can prove a representation theorem. Given the class of all models of a system, a representation theorem shows that every model is isomorphic to a member of a distinguished subset. For example, it is easy to see that some particular non-monotonic theories may be represented by neural networks. Consider the system discussed above for reasoning about birds: 2 input nodes (one for $Bird(x)$ and one for $Penguin(x)$) and an output node (for $Fly(x)$) are all we need to model this system with a simple neural network. So long as there is an excitatory connection from $Bird(x)$ to $Fly(x)$ and an even stronger inhibitory connection from $Penguin(x)$ to $Fly(x)$, this network will produce the same conclusions from the same premises as our non-monotonic theory. But this is just a specific case; a representation theorem for non-monotonic logics in neural networks would show us that for every non-monotonic theory, there is some neural network which computes the same conclusions. Such a theorem would demonstrate a deep computational equivalence between non-monotonic logics and neural networks.

As it turns out, representation theorems of this form have been given by several logicians coming from a variety of backgrounds and motivations. Hölldobler and collaborators prove a representation theorem for logic programs, demonstrating that for any logic program P, a three layer, feed forward network can be found which computes P (Hölldobler and Kalinke, 1994; Hitzler et al., 2004). Pinkas (1995) provides similar results for a wider class of neural networks and penalty logic. (Penalty logic is a non-monotonic logic which weights conditionals with positive integers representing the "penalty" if that conditional is violated. Reasoning in penalty logic involves identifying the set of propositions which minimizes the overall penalty for a set of these weighted conditionals.) Hölldobler and Pinkas are both working within the artificial intelligence community and, consequently, their results are focussed on practical applications, with an emphasis on algorithms for performing the translation from symbolic to connectionist system (and, in the case of Pinkas, vice versa).

Stenning and van Lambalgen (2008) extend the results of Hölldobler in two directions. First, they consider logic programs with negation as default (a true non-monotonic logic); second, their representation theorem is informed by their psychological analysis of the Wason selection task and

related phenomena. Unlike the logicians working within A.I., Stenning and van Lambalgen explicitly address the neural (and psychological) plausibility of the neural networks in their representation theorem.

As it turns out, the most common versions of logic programming with negation as default are at least as strong as the system **CL** discussed above (Leitgeb, 2005, pp. 199–200). Furthermore, since they employ Horn clauses, they cannot distinguish **CL** from some stronger non-monotonic logics (Kraus et al., 1990, p. 200). What about the weaker system **C** discussed by Kraus et al.? Leitgeb (2003) proves representation theorems for each system introduced by Kraus et al. in distinguished classes of neural networks (see also Leitgeb, 2001). Leitgeb (2003) gives its results in terms of inhibition nets, where an inhibition net is a spreading activation neural network with binary (i.e. firing or non-firing) nodes and both excitatory and inhibitory connections. If there is no hierarchical structure to the net, it exhibits non-monotonic reasoning as weak as **C**. Leitgeb (2005) extends this result from the particular case of inhibition nets to a much more general dynamical systems framework. It turns out that hierarchical structure is closely tied to the stronger logic **CL** in many different dynamical systems (layered networks such as those considered by Hölldobler and Pinkas are hierarchical, for example).

Unlike the other results discussed here, Leitgeb takes pains to ensure his representation theorems subsume the distributed case. In particular, he allows for a proposition letter to be interpreted as a set of nodes in a neural network. From a philosophical standpoint, this result should raise questions for the debate between symbolic and connectionist approaches. Leitgeb has shown that any dynamical system performing calculations over distributed representations may be interpreted as a symbolic system performing non-monotonic reasoning. Correspondingly, it appears as if there is no substantive difference in representational or problem-solving power between symbolic and connectionist systems. However, this is an "in principle" result; Leitgeb does not offer algorithms for constructing interpretations or extracting symbolic systems from trained neural nets. Even if there is no difference in which problems symbolic and connectionist systems can represent and solve, there may be differences in how fast or efficiently they can solve interesting classes of problems. The following two sections address this question from a computational perspective, attempting to bridge the gap between computational feasibility and psychological plausibility.

4 Computational Complexity and The Invariance Thesis

With the development of programming practice it was discovered that there are computable problems for which we do not know any efficient algorithms.

Some problems require too much time or memory to be feasibly solved by a realistic computational device. Computational complexity theory investigates the resources (time, memory, etc.) required for the execution of algorithms and the inherent difficulty of computational problems (see e.g. Papadimitriou, 1993; Arora and Barak, 2009, or the paper by Ramanujam in this volume). This means that the theory does not deal directly with concrete algorithmic procedures, but instead studies the abstract computational properties of queries. Given a problem, features which hold for all possible solution algorithms can be investigated in a precise mathematical sense. This allows us to precisely distinguish those problems which have efficient solutions from those which do not.

This method for analyzing queries allows us to sort them into complexity classes. In particular, we want to identify efficiently solvable problems and draw a line between tractability and intractability. From our perspective the most important distinction is that between problems which can be computed in polynomial time with respect to the size of the problem, i.e. relatively quickly, and those which are believed to have only exponential time algorithmic solutions. The class of problems of the first type is called PTIME (P for short). Problems belonging to the second are referred to as NP-hard problems. Intuitively, a problem is NP-hard if there is no efficient algorithm for solving it. The only way to deal with it is by using brute-force methods: searching throughout all possible combinations of elements over a universe. In other words, NP-hard problems lead to combinatorial explosion.

Notice that this categorization is helpful only under the assumption that the complexity classes defined in the theory are essentially different. These inequalities are usually extremely difficult to prove. In fact, the most famous problem in complexity theory is of this form, namely the widespread assumption that P$\neq$NP. This is considered one of the seven most important open mathematical problems by the Clay Institute of Mathematics, who have offered a \$1,000,000 prize for its solution. As we said above, PTIME is the class of problems which can be computed by deterministic Turing machines in polynomial time. Speaking precisely, NP-hard problems are problems which are at least as difficult as problems belonging to the NPTIME (NP) class; this is the class of problems which can be computed by nondeterministic Turing machines in polynomial time. NP-complete problems are NP-hard problems belonging to NPTIME, hence they are intuitively the most difficult problems among the NPTIME problems. If we could show that any NPTIME-complete problem is PTIME computable, we would have demonstrated that P=NP. Whether this is possible or not is still an open question. However, the experience and practice of computational complexity theory allow us to reasonably assume that these two classes are different.

Before we move to more general considerations, let us consider an example. Many natural problems are computable in polynomial time, for instance calculating the greatest common divisor of two numbers or looking some-

thing up in a dictionary. However, we will focus here on a very important NP-complete problem, the satisfiability problem for classical propositional calculus (SAT). The problem is to decide whether a given classical propositional formula is not a contradiction. Let φ be a propositional formula with $p_1, \ldots, p_n$ distinct variables. Let us use the well-known algorithm based on truth-tables to decide whether φ has a satisfying valuation. How big is the truth-table for φ? The formula has n distinct variables occurring in it and therefore the truth-table has 2^n rows. If $n = 10$ there are 1,024 rows, for $n = 20$ there are already 1,048,576 rows and so on. In the worst case, to decide whether φ is satisfiable we have to check all rows. Hence, in such a case, the time needed to find a solution is exponential with respect to the number of different propositional letters of the formula. A seminal result of computational complexity theory states that this is not a property of the truth-table method but of the inherent complexity of the satisfiability problem. We have the following: SAT is NP-complete.

The previous paragraph tells us something about the relation between classical logic and computability theory: even very simple logics can define extremely difficult computational problems. How about non-classical logics; in particular, what do we know about the computational complexity of reasoning with non-monotonic logics? It turns out that typically the computational complexity of non-monotonic inferences is higher than the complexity of the underlying monotone logic. As an example, restricting the expressiveness of the language to Horn clauses allows for polynomial inference as far as classical propositional logic is concerned. However, this inference task becomes NP-hard when propositional default logic or circumscription is employed. This increase in complexity can be explained by pointing out that semantic definitions in many non-monotonic logics (including those discussed above) are based on fixed-point constructions or some kind of minimality requirement (see Cadoli and Schaerf, 1993, for a survey on the topic).

In the early days of computational complexity theory, the following thesis was formulated independently by Alan Cobham (1965) and Jack Edmonds (1965): The class of practically computable problems is identical to the PTIME class, that is, the class of problems which can be computed by a deterministic Turing machine in a number of steps bounded by a polynomial function of the length of a query. This thesis is accepted by most computer scientists (see e.g. Garey and Johnson, 1979). However, for the claim to make sense, we need some additional assumptions which will return us to the symbolic vs. connectionist debate.

As we said before, computational complexity theory is concerned with the inherent complexity of problems independent of particular algorithmic solutions and their implementations. The most common model of computation used in this theory is the Turing machine. However, to justify computational complexity distinctions, e.g. between tractable and intractable

problems, we need to demonstrate that they hold independent of any particular implementation. The Invariance Thesis (see e.g. van Emde Boas, 1990) states that, given a "reasonable encoding" of the input and two "reasonable machines", the complexity of computation of these machines on that input will differ by at most a polynomial amount. By "reasonable machine", we mean any type of deterministic Turing machine or any other realistic computing machine. The situation here is very similar to that of the Church-Turing Thesis; although we cannot prove this claim, the fact that it holds for all known realistic models of computation provides powerful support for it. In this case, some well known machines are ruled out; for example, nondeterministic Turing machines and quantum computers are not realistic in this sense. Assuming the Invariance Thesis, we get that a task is difficult if it corresponds to a function of a high computational complexity, independent of the computational devices we are working with, at least as long as they are reasonable.

A natural question for cognitive science is, do neural networks fall within the scope of the Invariance Thesis? In fact, despite popular claims to the contrary, they do. From the computational complexity perspective the difference between serial and parallel machines is insubstantial as far as tractability is concerned. Although some parallel machines can compute certain functions faster than serial Turing machines, the speed-up is never more than a polynomial amount of time. As long as we assume that the input to such a machine may grow larger than the number of parallel channels, this speed-up rapidly becomes irrelevant. Therefore, the difference between symbolic and connectionist computations is negligible from the tractability perspective (see van Rooij, 2008, particularly Section 6.6, for extended discussion).

The common belief in the Cobham-Edmonds Thesis stems from the practice of programmers. NP-hard problems often lead to algorithms which are not practically implementable even for inputs of not very large size. Assuming the Church-Turing Thesis, $P \neq NP$, and the Invariance Thesis, one comes to the conclusion that this has to be due to some internal properties of these problems and not to the limitations of current computing technology.

5 Applying Complexity in Cognitive Science

Accepting the Cobham-Edmonds Thesis we have to agree that problems beyond PTIME are computationally intractable, a restriction which applies to both symbolic and connectionist models. How does this conclusion influence cognitive science?

From an abstract perspective, a cognitive task is an information-processing or computational task. In general, the aim of a cognitive task is to transform the initial given state of the world into some desired final

state. Therefore, cognitive tasks can be identified with functions from possible initial states of the world into possible final states of the world. Notice that this understanding of cognitive tasks is very closely related to psychological practice. First of all, experimental psychology is naturally task oriented, because subjects are typically studied in the context of specific experimental tasks. Furthermore, the dominant approach in cognitive psychology is to view human cognition as a form of information processing (see e.g. Sternberg, 2002).

One of the primary objectives of behavioral psychology is to explain human cognitive tasks as understood in the abstract sense outlined here. David Marr (1983) was the first to propose a commonly accepted general framework for explanation in cognitive science. Any particular task computed by a cognitive system must ultimately be analyzed at three levels (in order of decreasing abstraction): (1) the computational level (the problem solved or function computed); (2) the algorithmic level (the algorithm used to achieve a solution); (3) the implementation level (how the algorithm is actually implemented in neural activity). Considerations at each of these levels may constrain answers at the others; however, Marr argues that analysis at the computational level is the most critical for achieving progress in cognitive science (Marr, 1983, p. 27).

One of the abstract properties of tasks specified at Marr's computational level is their complexity. Since we do not know the details of our cognitive hardware or the precise algorithms implemented in the brain, the inherent perspective of computational complexity theory is well-suited for a general investigation of the plausibility of computational task analyses. For example, conclusions from complexity analysis are invariant with respect to the results of the philosophical debate between advocates of symbolic and connectionist models in cognitive science. Moreover, by studying the computational complexity of a problem, we can provide knowledge about the simplest possible means of solving it. In this sense, complexity analysis lies between Marr's first two levels (though it should not be confused with the level 1.5 of Peacocke, 1986).

One of the most common computational claims about cognition is a psychological version of the Church-Turing Thesis: the human mind can only deal with computable problems. In other words, cognitive tasks comprise computable functions. Despite its widespread acceptance, the psychological version of the Church-Turing Thesis has its critics. The first source of opposition is those who believe that cognitive systems can do more than Turing machines. For example, learning understood as identifiability in the limit (Gold, 1967) is not computable (see Kugel, 1986, for an extensive discussion), yet it plausibly falls within the scope of cognitive science. Another strand of opposition arises from concerns about practical computability. Since cognitive systems are physical systems, they perform tasks under computational resource constraints. Therefore, the functions computed by cognitive sys-

tems need to be computable in realistic time and with the use of a realistic amount of memory. We agree with this objection, and will consider it in more detail.

The worry that plausible models of agents in a complex world must take into account the limits of their computational resources is often called the problem of bounded rationality (Simon, 1957, and many following publications). Simon argued that the limited computational resources of bounded agents require them to solve many problems with rough heuristics rather than exhaustive analyses of the problem space. In essence, rather than solve the hard problem presented to him by the environment, the agent solves an easier, more tractable problem which nevertheless produces an acceptable solution. In order to apply this insight in cognitive science, it would be helpful to have a precise characterization of which class of problems can plausibly be computed by the agent. The answer suggested by complexity theory is to adapt the Cobham-Edmonds Thesis: The class of problems which can be computed by a cognitive agent is approximated by the PTIME class, i.e. bounded agents can only solve problems with polynomial time solutions. As far as we are aware, the version of the Cobham-Edmonds Thesis for cognitive science was first formulated explicitly in print by Frixione (2001) and later dubbed the P-Cognition Thesis by van Rooij (2008). The P-Cognition Thesis states that a cognitive task is (hard) easy if it corresponds to a(n) (in)tractable problem.

The P-Cognition Thesis can be used to analyze which problem an agent is plausibly solving when the world presents him with an (apparently) intractable problem. For example, Levesque (1988) argues that the computational complexity of general logic problems motivates the use of Horn clauses and other tractable formalisms to obtain psychologically realistic models of human reasoning. Similarly, Tsotsos (1990) emphasizes that visual search in its general (bottom-up) form is NP-complete. As a consequence, only visual models in which top-down information constrains visual search space are computationally plausible. In the study of categorization and subset choice, computational complexity serves as a good evaluation of psychological models (see e.g. van Rooij et al., 2005). Recently, Szymanik (2009) has applied computational complexity analysis to the psycholinguistic study of the meaning of quantifiers in natural language (see also Szymanik and Zajenkowski, 2010). This general strategy for analyzing cognitive tasks can also be found in philosophy of mind; for example, Cherniak (1981) argues that tractability considerations demand a philosophical analysis of the conditions required for a minimal notion of rationality.

Still, our experience shows that many researchers are skeptical about the applicability of computational complexity in cognitive modeling. Therefore, we conclude by addressing some common objections.

Computational complexity is defined in terms of limit behavior. In other words, a typical question of computational complexity theory is of the form:

As the size of the input increases, how do the running time and memory requirements of the algorithm change? Therefore, computational complexity theory, among other things, investigates the scalability of computational problems and algorithms, i.e. it measures the rate of increase in computational resources required as a problem grows. The implicit assumption here is that the size of the problem is unbounded. For example, models can be of any finite size, formulae can contain any number of distinct variables, and so on.

In general, even though computational complexity is formally defined in terms of limit behavior, it can still be reasonably interpreted as saying something about problem difficulty on a fixed model. Namely, if the computational complexity of the problem is high, then it means that there are no "clever" algorithms for solving it, i.e. we must perform an exhaustive search through the entire solution space. Therefore, it is very likely that on a given fixed model we also have to use a brute-force method, and this will be again difficult (even for relatively small n). For example, checking SAT for a formula containing 5 variables is already quite time-consuming. Moreover, even if typical cognitive situations involve reasonably small inputs, it would be difficult to justify any *in principle* bound on their size. Potentially, inputs can grow without limit. Therefore, computational complexity may reasonably be proposed as a difficulty measure for tasks considered at the abstract level—a standard and usually fruitful idealization in science.

This abstract perspective can be constructively applied in experimental contexts. For example, differences in performance (e.g. reaction time) on an experimental task may be used to fruitfully analyze the computation the agent is performing. We can even track the changes in heuristic a single agent employs as the problem space changes. For example, it is known that reaction time increases linearly when subjects are asked to count between 4 and 15 objects. Up to 3 or 4 objects the answer is immediate, so-called subitizing. For judgments involving more than 15 objects, subjects start to approximate: reaction time is constant and the number of incorrect answers increases dramatically (Dehaene, 1999). Analogously, we can ask what gross differences in reaction time on NP-hard tasks say about the nature of the heuristics subjects employ. A particularly telling example here is the case of chess experts. The radical increase in reaction time of chess experts over that of novices does not necessarily indicate that they are solving the same (NP-hard) problem of exhaustively searching the tree of possible moves, but rather that they are solving the different (and much simpler) problem of searching only the "good" moves available to them (Simon and Simon, 1962). The P-Cognition Thesis guides our analysis of expert behavior here, suggesting that the algorithms experts implement for constraining search may be PTIME in complexity.

Another criticism of the P-Cognition Thesis questions the value of worst-case computational complexity as a measure of difficulty for cognitive pro-

cesses. The point here is whether we really need to consider the worst-case performance of cognitive algorithms on arbitrary inputs. It might be the case that inputs generated by nature have some special properties which make problems tractable on those inputs even though in principle they might be NP-hard. The natural strategy here is to turn toward so-called average-case complexity theory. It studies the computational complexity of problems on randomly generated inputs. The theory is motivated by the fact that many NP-hard problems are in fact easily computable on "most" of the inputs. But average-case complexity theory extends and supplements the worst-case analysis. It does not, in general, replace it. For example, if the appropriate probability distribution over nature's behavior is unavailable, average-case complexity analysis simply cannot be applied. Furthermore, it seems that the worst-case analysis is actually the right perspective from which to analyze cognitive tractability. Average-case complexity may still be preferred for purposes other than assessing tractability, e.g. comparing the time-complexity of different (tractable) algorithmic-level explanations with reaction time data obtained via experimentation (see van Rooij, 2008).

Another strategy for cognitively plausible complexity measures is to break up each task into parameters and analyze how each of the parameters contribute to the overall complexity. It might be the case that intractability of some problems comes from a parameter which is usually very small no matter how big the input. This way of thinking leads to the consideration of parametrized complexity theory as a measure for the complexity of cognitive processes. Iris van Rooij (2008) investigates this subject, proposing the Fixed-Parameter Tractability Thesis as a refinement of the P-Cognition Thesis.

One of the great challenges in understanding the mind as a computational device is determination of the correct complexity measures. We have argued that computational complexity theory provides the right measure of cognitive tractability. Furthermore, the Invariance Thesis implies that the correct notion of tractability will not depend upon the details of one's computational model, and the P-Cognition Thesis implies that only PTIME problems are tractable. Consequently, even if symbolic and connectionist modelers diverge in their analyses at Marr's algorithmic level, they should agree on the correct analysis at the computational level. Although complexity theory can be supplemented in various ways (by considering only average-cases or by fixing parameters), these refinements do not obviate the need for an analysis of cognitive tractability. Finally, complexity analysis can be used to bridge the gap between abstract task analysis and empirical data by providing testable predictions about how changes in input size or training affect reaction times.

6 Conclusion

We began with a brief overview of the computational perspective on cognitive science. We saw that logical techniques constitute a common language for many of the diverse fields studying cognition. One particularly important tool here is non-monotonic logic, which has helped the artificial intelligence community to better understand and model common sense reasoning. On a more abstract level, a series of recent results seems to show that non-monotonic logics and neural networks are in some deep sense equivalent. This result is surprising given famous arguments from the philosophy of mind that there is a fundamental difference in the representational properties of neural networks and symbolic systems. Worried that our analysis was simply too coarse to capture this fundamental difference, we turned to a discussion of the computational complexity of the two systems. However, the Invariance Thesis indicates that there is no *in principle* distinction between neural nets and symbolic systems from a complexity perspective, either. The distinctive features of parallel computation in neural networks are entirely contingent (depending upon bounds on input size, or the number of available parallel channels, neither of which can be motivated at the abstract level of computational task analysis).

Finally, we defended this abstract level of analysis as the appropriate one for determining cognitive tractability. The P-Cognition Thesis posits that only PTIME problems are tractable for cognitive agents. Perhaps surprisingly, this very abstract claim nevertheless connects closely to empirical practice. Differences in reaction time across changes in experimental condition or training can help us detect changes in how bounded agents interpret and solve problems in a complex world. The P-Cognition Thesis motivates restricting our task analysis at Marr's computational level to functions computable in polynomial time. From this perspective, symbolic and connectionist modelers should agree at the computational level, with any disagreements emerging only at the algorithmic level. Conversely, sharp differences in computational complexity may be taken as evidence that two modelers are in fact analyzing distinct tasks. And it is here that the tools of psychology must take over from those of logic, and the models under investigation be subjected to renewed empirical tests.

Acknowledgments

We would like to thank Johan van Benthem, Iris van Rooij, and Keith Stenning for many comments and suggestions. The first author would also like to thank Douwe Keila and Thomas Icard for many helpful discussions of this material. The second author was supported by a postdoctoral research grant funded by the Swedish Research Council.

References

Anderson, J. R. and Lebiere, C. (1998). *The Atomic Components of Thought*. Erlbaum, Mahwah, NJ.

Arora, S. and Barak, B. (2009). *Computational Complexity: A Modern Approach*. Cambridge University Press, 1 edition.

Cadoli, M. and Schaerf, M. (1993). A survey of complexity results for non-monotonic logics. *J. Log. Program.*, 17(2/3&4):127–160.

Chalmers, D. (1990). Why Fodor and Pylyshyn were wrong: The simplest refutation. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, pages 340–347.

Cheng, P. W., Holyoak, K. J., Nisbett, R. E., and Oliver, L. M. (1986). Pragmatic vs. syntactic approaches to training deductive reasoning. *Cognitive Psychology*, 18:293–328.

Cherniak, C. (1981). Minimal rationality. *Mind*, 90(358):161–183.

Clark, A. (1993). *Associative Engines*. Bradford Books, Cambridge, MA.

Clark, H. (1976). *Semantics and Comprehension*. Mouton.

Clark, K. L. (1978). Negation as failure. In Gallaire, H. and Minker, J., editors, *Logics and Data Bases*. Plenum Press, New York, NY.

Clark, R. and Grossman, M. (2007). Number sense and quantifier interpretation. *Topoi*, 26(1):51–62.

Cobham, J. (1965). The intrinsic computational difficulty of functions. In *roceedings of the 1964 International Congress for Logic, Methodology, and Philosophy of Science*, pages 24–30. North-Holland.

Cooper, B. S. (2003). *Computability Theory*. Chapman Hall/Crc Mathematics Series. Chapman & Hall/CRC.

Cosmides, L. (1989). The logic of social exchange: Has natural selection shaped how humans reason? Studies with the Wason selection task. *Cognition*, 31:187–276.

Cosmides, L. and Tooby, J. (1992). Cognitive adaptations for social exchange. In Barkow, J., Cosmides, L., and Tooby, J., editors, *The adapted mind: evolutionary psychology and the generation of culture*, pages 163–228. Oxford UP, Oxford, UK.

Dehaene, S. (1999). *The Number Sense: How the Mind Creates Mathematics*. Oxford University Press, USA.

Dennett, D. C. (1984). Cognitive wheels: The frame problem of AI. In Hookway, C., editor, *Minds, Machines and Evolution: Philosophical Studies*. Cambridge UP, Cambridge.

Edmonds, J. (1965). Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467.

Feldman, J. (2000). Minimization of Boolean complexity in human concept learning. *Nature*, 407(6804):630–633.

Flobbe, L., Verbrugge, R., Hendriks, P., and Krämer, I. (2008). Children's application of theory of mind in reasoning and language. *Journal of Logic, Language and Information*, 17(4):417–442.

Fodor, J. A. (1983). *The Modularity of Mind: An Essay on Faculty Psychology*. MIT Press, Cambridge, MA.

Fodor, J. A. and Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: a critical analysis. *Cognition*, 28:3–71.

Frixione, M. (2001). Tractable competence. *Minds and Machines*, 11(3):379–397.

Gärdenfors, P. (2000). *Conceptual Spaces: The Geometry of Thought*. MIT Press, Cambridge, MA.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability*. W. H. Freeman and Co., San Francisco.

Geurts, B. (2003). Reasoning with quantfiers. *Cognition*, 86:223–251.

Geurts, B. (2010). *Quantity Implicatures*. Cambridge University Press.

Gierasimczuk, N. and Szymanik, J. (2009). Branching Quantification vs. Two-way Quantification. *Journal of Semantics*, 26(4):367–392.

Gold, E. M. (1967). Language identification in the limit. *Information and Control*, 10:447–474.

Griggs, R. A. and Cox, J. R. (1982). The elusive thematic-materials effect in Wason's selection task. *British Journal of Psychology*, 73:407–420.

Hitzler, P., Hölldobler, S., and Seda, A. K. (2004). Logic programs and connectionist networks. *Journal of Applied Logic*, 2(3):245–272. Neural-symbolic Systems.

Hölldobler, S. and Kalinke, Y. (1994). Toward a new massively parallel computational model for logic programming. In *Proc. Workshop on Combining Symbolic and Connectionist Processing, ECAI-94*, Amsterdam.

Immerman, N. (1999). *Descriptive complexity*. Springer Verlag.

Johnson-Laird, P. (2008). *How we reason*. Oxford University Press.

Johnson-Laird, P. N., Legrenzi, P., and Legrenzi, M. S. (1972). Reasoning and a sense of reality. *British Journal of Psychology*, 63:395–400.

Kraus, S., Lehmann, D. J., and Magidor, M. (1990). Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence*, 44(1–2):167–207.

Kugel, P. (1986). Thinking may be more than computing. *Cognition*, 22(2):137–198.

Leitgeb, H. (2001). Nonmonotonic reasoning by inhibition nets. *Artificial Intelligence*, 128(1–2):161–201.

Leitgeb, H. (2003). Nonmonotonic reasoning by inhibition nets II. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 11(2 (Supplement)):105–135.

Leitgeb, H. (2005). Interpreted dynamical systems and qualitative laws: From neural networks to evolutionary systems. *Synthese*, 146:189–202.

Levesque, H. J. (1988). Logic and the complexity of reasoning. *Journal of Philosophical Logic*, 17(4):355–389.

Marr, D. (1983). *Vision: A Computational Investigation into the Human Representation and Processing Visual Information*. W.H. Freeman, San Francisco.

McCarthy, J. (1980). Circumscription—a form of non-monotonic reasoning. *Artificial Intelligence*, 13(1–2):27–39.

McCarthy, J. and Hayes, P. J. (1969). Some philosophical problems from the standpoint of artificial intelligence. In Michie, D. and Meltzer, B., editors, *Machine Intelligence*, volume 4, pages 463–502. Edinburgh University Press.

McCulloch, W. S. and Pitts, W. H. (1943). A logical calculus immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133.

Moschovakis, Y. (2006). A Logical Calculus of Meaning and Synonymy. *Linguistics and Philosophy*, 29(1):27–89.

Moxey, L. and Sanford, A. (1993). *Communicating Quantities. A psychological perspective*. Lawrence Erlbaum Associates Publishers.

Papadimitriou, C. H. (1993). *Computational Complexity*. Addison Wesley.

Peacocke, C. (1986). Explanation in computational psychology: Language, perception and level 1.5. *Mind and Language*, 1:101–23.

Peters, S. and Westerståhl, D. (2006). *Quantifiers in Language and Logic*. Clarendon Press.

Pinkas, G. (1995). Reasoning, nonmonotonicity and learning in connectionist networks that capture propositional knowledge. *Artificial Intelligence*, 77:203–247.

Reiter, R. (1980). A logic for default reasoning. *Artificial Intelligence*, 13:81–132.

van Rooij, I. (2008). The tractable cognition thesis. *Cognitive Science: A Multidisciplinary Journal*, 32(6):939–984.

Sandler, U. and Tsitolovsky, L. (2008). *Neural Cell Behavior and Fuzzy Logic*. Springer, New York.

Shanahan, M. (1997). *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia*. MIT Press, Cambridge, MA.

Simon, H. A. (1957). *Models of Man: Social and Rational*. John Wiley and Sons, Inc., New York, NY.

Simon, H. A. and Simon, P. A. (1962). Trial and error search in solving difficult problems: Evidence from the game of chess. *Behavioral Science*, 4(2):425–429.

Smolensky, P. (1987). Connectionism and cognitive architecture: a critical analysis. *Southern Journal of Philosophy*, 26 (supplement):137–63.

Smolensky, P. (1988). On the proper treatment of connectionism. *Behavioral and Brain Sciences*, 11:1–74.

Smolensky, P. (1991). Connectionism, constituency, and the language of thought. In Loewer, B. and Rey, G., editors, *Meaning in Mind: Fodor and His Critics*, pages 201–227. Blackwell.

Stenning, K. and van Lambalgen, M. (2008). *Human Reasoning and Cognitive Science*. MIT Press, Cambridge, MA.

Sternberg, R. J. (2002). *Cognitive Psychology*. Wadsworth Publishing.

Suppes, P. (1982). Variable-Free Semantics with Remarks on Procedural Extensions. In Simon, T. and Scholes, R., editors, *Language, Mind, and Brain*, pages 21–34. NJ:Erlbaum, Hildsdale.

Szymanik, J. (2007). A comment on a neuroimaging study of natural language quantifier comprehension. *Neuropsychologia*, 45:2158–2160.

Szymanik, J. (2009). *Quantifiers in TIME and SPACE. Computational Complexity of Generalized Quantifiers in Natural Language*. PhD thesis, Universiteit van Amsterdam.

Szymanik, J. and Zajenkowski, M. (2010). Comprehension of simple quantifiers. Empirical evaluation of a computational model. *Cognitive Science*, 34(3):521–532.

Tsotsos, J. (1990). Analyzing vision at the complexity level. *Behavioral and Brain Sciences*, 13(3):423–469.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59:433–460.

van Emde Boas, P. (1990). Machine models and simulations. In van Leeuwen, J., editor, *Handbook of Theoretical Computer Science*, pages 1–66. MIT Press, Cambridge, MA, USA.

van Gelder, T. (1990). Compositionality: A connectionist variation on a classical theme. *Cognitive Science*, 14:355–364.

van Gelder, T. (1991). Classical questions, radical answers: Connectionism and the structure of mental representations. In Horgan, T. and Tienson, J., editors, *Connectionism and the Philosophy of Mind*, pages 355–381. Kluwer Academic Publishers.

van Harmelen, F., Lifschitz, V., and Porter, B., editors (2008). *Handbook of Knowledge Representation*. Elsevier, Amsterdam.

van Lambalgen, M. and Hamm, F. (2004). *The Proper Treatment of Events*. Blackwell.

van Rooij, I., Stege, U., and Kadlec, H. (2005). Sources of complexity in subset choice. *Journal of Mathematical Psychology*, 49(2):160–187.

Verbrugge, R. (2009). Logic and social cognition. *Journal of Philosophical Logic*, 38(6):649–680.

Verbrugge, R. and Mol, L. (2008). Learning to apply theory of mind. *Journal of Logic, Language and Information*, 17(4):489–511.

Wason, P. C. (1968). Reasoning about a rule. *Quarterly Journal of Experimental Psychology*, 20:273–281.

Wason, P. C. and Shapiro, D. (1971). Natural and contrived experience in a reasoning problem. *Quarterly Journal of Experimental Psychology*, 23:63–71.